Security Testing as part of Software Quality Assurance: Principles and Challenges

Wissam Mallouli
Research & Development Dept.
Montimage EURL
Paris, France
wissam.mallouli@montimage.com

Software quality assurance (SQA) is a means and practice of monitoring the software engineering processes and methods used in a project to ensure proper quality of the software. It encompasses the entire software development life-cycle, including requirements engineering, software design, coding, source code reviews, software configuration management, **testing**, release management, software deployment and software integration. It is organized into goals, commitments, abilities, activities, measurements, verification and validation. In this talk, we will mainly focus on the testing activity part of the software development life-cycle. Its main objective is checking that software is satisfying a set of quality properties that are identified by the "ISO/IEC 25010:2011 System and Software Quality Model" standard [1].

Indeed, this ISO/IEC 25010 standard identified eight software quality characteristics to assess its quality. These characteristics are functional suitability, reliability, performance efficiency, use-ability, **security**, compatibility, maintainability and portability.

Security refers to how well a software or system protects information and data from security vulnerabilities. The complexity and the variety of the deployed time dependent systems, as well as the high degree of security and robustness required for their global functioning, justify the care provided to the design of the best possible tests. Moreover, it is significant to automate these steps with an aim of reducing the time and the development cost and especially of increasing the quality and security of the offered products.

To reach this objective, different security testing techniques has been conceived in the literature [2] and most of them are used in classical technical security audit of IT systems. They either rely on predefined security requirements or on identified threats (part of risk assessment), and in the majority of cases on both of them. They use different techniques like:

- Model based testing : in general the system under test is specified as well as its security requirements and/or identified threats.
- Risk based testing : a risk assessment is performed to identify vulnerabilities that can be exploited by testers.
- Data based testing : data is mutated to check its impact on the system (e.g., malformed data or bad data types).
- Mutation testing : Like fuzz testing, it uses different mutation operators and combine them to generate malicious and unexpected inputs.
- ML based testing : The optimization of tests suites as well as their coverage can be performed using machine learning techniques.

Security testing techniques also depend on our knowledge level of the system/software under test. White box testing methods considers that internal structure as well as source code are known to tester which is not the case for black box testing methods. Static and Dynamic code analysis are thus performed using different tools and techniques. The big challenges for software security testing are in generals as follows:

- High-priority vulnerabilities: It is not allowed to make trade-offs in resources and coverage while performing security testing. A vulnerability (even minor) can have a big impact of the whole application and system resiliency.
- Test hidden parts of the application: Understanding the relationship between application components and data flows is not always easy when performing black box security testing.
- Security testing during operation: This task requires maintenance to perform tests that can harm the availability of the running application.
- Available interfaces: Some software or system does not allow external interactions which makes its testing very hard or requiring specific hardware components.
- Software context: the context can be itself vulnerable generating thus more risks.
- Tackling zero days vulnerabilities and attacks.

ACKNOWLEDGEMENT

This work was made possible with funding from the European Union's Horizon 2020 research and innovation programme, under grant agreement No. 957212 (VeriDevOps). The opinions expressed and arguments employed herein do not necessarily reflect the official views of the funding body.

REFERENCES

[1] ISO/IEC 25010, *ISO/IEC 25010:2011, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, Std., 2011.

[2] M. Felderer, M. Büchler, M. Johns, A. D. Brucker, R. Breu, and A. Pretschner, "Security testing: A survey," *Adv. Comput.*, vol. 101, pp. 1–51, 2016. [Online]. Available: https://doi.org/10.1016/bs.adcom.2015.11.003